REMARKS

Introduction

Claims 1-17 are pending. Claims 1, 11, and 17 are independent. Claims 1, 4-8, 11, and 17 have been amended.

Rejections under 35 U.S.C. § 112

Claims 1 and 11 stand rejected under 35 U.S.C. § 112, first paragraph. The Examiner stated that "Java-like" interface is not mentioned in the specification and that claim 1 and 11 use the term "Java-like interface" while claim 17 refers to a "Java interface." Claims 1, 11, and 17 have been amended to recite an "abstract interface," which is a term known in the art as referring to a data structure which contains a set of unpopulated function signatures. A given class, in this case the data set structure, promises to implement or populate the member functions of the interface so that a call to an interface member function invokes underlying legitimate class code. Support for "abstract interface" can be found at least in FIG 2 wherever the term <<interface>> is shown, and in paragraph [0017] of the application as published: U.S. Patent Application Publication No. 2004/0230602.

Rejections under 35 U.S.C. § 102(e)

Claims 1-17 stand rejected under 35 U.S.C. 102(e) as unpatentable over U.S. Patent Application Publication No. 2003/0078960 (Murren et al.).

Murren et al. describes a multi-layer software architecture for the construction of business processes and server-based software applications for various business domains. The architecture is arranged into several hierarchical layers. An execution environment layer handles

incoming requests from remote clients and selects the appropriate problem-solving logic in a business logic layer to process the requests. A presentation layer structures the replies generated by the business logic layer into a desired appearance and encodes the replies using formats and communication protocols supported by different clients. Any one of the layers may be removed, modified, or updated without impacting other layers. The architecture supports a hierarchy of constraint layers, where each layer imposes different business constraints on how the application might operate or how content may be presented to the user (see paragraph [0161]). A set of constraints (understood to mean limitations) are placed on various configuration parameters and application functions of the application.

Amended claim 1 and 17 of the present application recite a method, while claim 11 recites a data structure, implemented by the method claim 1. Each of claims 1, 11, and 17 recite a data set-structure which implements an abstract interface for use in both the business layer and the presentation layer, the data set structure comprising hierarchical organizational information for arranging one of data and functions into at least one tree structure, the true structure being navigable without regard to the type of data or function being processed; populating a business layer data set in said business layer according to the data set-structure, the business layer data set comprising data and functions available for use in the business layer; and populating a presentation layer data set in the presentation layer according to the data set structure from the business layer data set, the presentation layer data set comprising data and functions available for use by the user in said presentation layer.

The claimed data set structure of the present application provides a uniform code interface for access to data items and function items, which ultimately store data structures and functions respectively. Data sets are defined and accessed in both the business layer and the

presentation layer to provide a uniform interface for functions and data regardless of the type of data structure and function. Data sets, along with data items and function items which encapsulate data and functions, respectively, constitute a modified version of the Composite pattern as described in E. Gamma, R. Helm, R Johnson, and J. Vlissides, "Design Patterns," Addison Wesley, 1995, pp. 163-173 (See paragraph [0018]). In a composite design pattern, objects (data and functions) are composed into tree structures (a hierarchy of functions and data) to represent part-whole hierarchies. As is known in the art, composites allow clients (business layer or presentation layer calls) to treat individual objects and compositions uniformly. Tree structure-like class hierarchies comprising simple or complex data structures and functions can be composed into more complex objects, which in turn can be composed, and so on recursively. Thus, when new structures and functions need to be added in the business layer or the presentation layer, the new functions maintain the common data set interface. Underlying code can be added in the business later and presentation layer at will, but the invoker of the function or call to data see the same interface, i.e. the data set interface. The data or functions can be added and/or stored in the tree structure set up by a user employing the data set interface. These tree structures can be traversed to access the appropriate data structure or function call regardless of the underlying type of data or function call.

As described above, Murren et al. does not describe a data set structure. Murren et al. describes two types of hierarchies: a hierarchy of modular layers, such as the business logic layer and the presentation layer, and a constraint hierarchy. The business logic layer performs functions equivalent to the user layer and business layer of the present claimed invention, while the presentation layer of Murren et al. takes raw data from the business layer and formats it for view by a user in a form desired by the user. As described by Murren et al., at paragraph [0051],

in Murren et al., the business logic layer is application-specific and is written on a per-application basis. Murren et al. does not specify how or whether the presentation layer is modified when code changes in the business layer, except to say that when the business layer logic changes, suitable modules can be configured in the presentation layer (see paragraph [0046]). At paragraph [0152] of Murren et al., the presentation layer is described as implemented as a dual tier layer which may contain a proprietary protocol or application programming interface (API), but does not specify the design of the API in terms of functions, classes, and abstract interfaces and their relationships as the presented invention describes in amended claims 1, 11, and 17. These changes are not described to be the design of classes/interfaces that are configured as data and function type independently and stored and accessible in the form of tree structures as provided by a data set structure. There is no mention Murren et al. of a data set structure-like interface that can be used in both the business layer and the presentation layer.

The other type of hierarchy described by Murren et al. refers to a hierarchy of business logic constraints that can be imposed on the presentation layer. This hierarchy refers to business applications, not hierarchies of functions and data structures. The constraints are rules that are placed on the format of output data as output by the presentation layer. These rules are based on legal, government, company, customer, and user constrains, etc. They are thus requirements for design of the software, not a hierarchy of an already designed software architecture as enabled by a data set structure, which provides information for configuring data and functions into tree structures that are independent of the data type or function type.

Accordingly, applicant submits that Murren et al. does not describe or teach each and every limitation of the invention recited by claims 1, 11, and 17 of the present application,

and withdrawal of the rejection of claim 1, 11, and 17 under 35 U.S.C. 102(e) based on Murren et al. is requested.

Each of claims 2-10 ultimately depend from claim 1, and claims 12-16 ultimately depend from claim 11. Since claims 1 and 11 have been shown to be patentable, each of claims 2-10 and 12-16 are patentable, for at least the reasons described above with respect to the patentability of claims 1 and 11.

Thus, applicants submit that each of the claims of the present application are patentable over each of the references of record, either taken alone, or in any proposed hypothetical combination. Accordingly, withdrawal of the rejections to the claims is respectfully requested.

Conclusion

In view of the above remarks, reconsideration and allowance of the present application is respectfully requested. No fee is believed to be due in connection with this Amendment. If, however, other fees are deemed necessary for this Amendment to be entered and considered by the Examiner, then the Commissioner is authorized to charge such fee to Deposit Account No. 50-1358. Applicant's undersigned patent agent may be reached by telephone at (973) 597-2500. All correspondence should continue to be directed to our address listed below.

Respectfully submitted,

For Raymond G. Cappo
Reg. No. 53,836

By: / James F. Dobrow /
James F. Dobrow
Attorney for Applicant
Registration No. 46,666

Date: 30 August 2007

DOCKET ADMINISTRATOR
LOWENSTEIN SANDLER PC
65 Livingston Avenue
Roseland, NJ 07068